Jawaban

1. Sebutkan library apa saja yang dipakai, website library itu dimana, dan dokumentasi library itu ada dimana.

Jawab:

Library Javascript utama yang dipakai adalah Backbonejs website dan dokumentasi library tersebut dapat langsung diakses ke home page library tersebut di https://backbonejs.org/.

Sedangkan library Javascript pendukung lainya yang digunakan yaitu:

- jQuery (alamat website: https://jquery.com/) (Dokuementasi: https://api.jquery.com)
- Underscore (alamat website dan dokumentasi https://underscorejs.org/)
- jQueryUI (alamat website: https://jqueryui.com/) (Dokumentasi: https://api.jqueryui.com/)

Untuk Library CSS yang digunakan pada aplikasi itu adalah bootstrap alamat website https://getbootstrap.com/ untuk dokumentasinya bisa diakses https://getbootstrap.com/docs/4.4/getting-started/introduction/.

2. Aplikasi itu 'laggy'. Kenapa? Bagaimana cara membuat animasi lebih 'smooth'?

Jawab:

Aplikasi terasa 'laggy' dikarenakan delay pada animasi pada setiap kata itu memiliki interval delay sebesar 100ms atau 328 fps, sedangkan prinsip animasi agar animasi tersebut smooth di sekitar 60fps atau 18ms.

Sehingga kita tinggak mengubah interval delay pada animasi menjadi 18ms, seperti kode dibawah ini

```
var animation_delay = 18; // rubah delaynya
setInterval(function() {
  self.iterate();
}, animation_delay);
```

3. Aplikasi itu tidak akan jalan di salah satu 3 browser populer (Chrome, Firefox, Internet Explorer)? Kenapa? Solusinya hanya menghapus satu character di code, character yang mana?

Jawab:

Untuk pertanyaan ke 3 saya mendapatkan kesempatan untuk melakukan skip atau mengabaikan , setelah saya bertanya dengan HRD, dan juga aplikasi dapat saya jalankan di broswer yang saya pakai version 79.0.3945.130 (Official Build) (64-bit), Firefox version 72.0, dan Internet Explorer version 11.1.19041.

4. Implementasikan tombol Start, Stop, Pause, dan Resume.

Jawab:

Saya mengimplementasikan tombol Start, Stop, Pause, dan Resume ke View dengan variable dengan nama GameControlView dan menambahkan method start, stop, pause pada model Typer .

```
// typer.js
// ...
var GameControlView = Backbone.View.extend({
  template: _.template(`
    <div class="game-control">
      <button id="start">start</button>
      <button id="pause">pause</button>
      <button id="resume">resume</button>
      <button id="stop">stop</button>
    </div>
  `),
  render: function() {
    $(this.el).append(this.template(this.model.toJSON()));
    return this;
  },
  initialize: function() {
    $(".form-control").attr("disabled", true);
    this.render();
    handleDisableButton(["start"], $, "stop");
  },
  events: {
    "click #start": "handleStart",
    "click #pause": "handlePause",
    "click #resume": "handleResume",
    "click #stop": "handleStop"
  },
  handleStart: function() {
    if (checkStatusJalan(this, ["STOP"])) {
      this.model.start();
      enableInput();
      handleDisableButton(["stop", "pause"], $, "start");
    }
  },
  handlePause: function() {
    if (!checkStatusJalan(this, ["START", "RESUME"])) return;

    this.model.pause();
    disableInput();
    handleDisableButton(["resume"], $, "pause");
  },
  handleResume: function() {
    if (checkStatusJalan(this, ["PAUSE"])) {
      this.model.resume();
      enableInput();
      handleDisableButton(["pause", "stop"], $, "resume");
    }
  },
  handleStop: function() {
    if (!checkStatusJalan(this, ["START", "RESUME"])) return;

    this.model.stop();
    disableInput();
```

```
      handleDisableButton(["start"], $, "stop");
    }
  });
  // ....
```

```
// ... Typer Model
    start: function() {
    // animatiaon_delay berpengaru terhadap fps 60fps === 16ms
    var animation_delay = 18;
    var self = this;
    const iterateON = setInterval(function() {
      self.iterate();
    }, animation_delay);
    this.set({ iterateON, statusGame: "START", skor: 0 });
  },

  resume: function() {
    // animatiaon_delay berpengaru terhadap fps 60fps === 16ms
    var animation_delay = 18;
    var self = this;
    const iterateON = setInterval(function() {
      self.iterate();
    }, animation_delay);
    this.set({ iterateON, statusGame: "RESUME" });
  },

  pause: function() {
    const iterateON = this.get("iterateON");
    clearInterval(iterateON);
    this.set({ statusGame: "PAUSE" });
  },

  stop: function() {
    const iterateON = this.get("iterateON");
    clearInterval(iterateON);
    const words = this.get("words");
    words.models.forEach(model => {
      model.get("view").remove();
    });
    this.set({ statusGame: "STOP", skor: 0 });
  },
```

5. Ketika ukuran window dirubah, susunan huruf yang 'terbentur' batas window menjadi tidak 1 baris. Benarkan.

Jawab:

saya memberikan event pada window saat ukuran window dirubah dimana akan melakuakan pengecekan apabila susunan huruf "terbentur" maka posisi huruf akan bergeser sehingga menjadi satu baris lagi.

```
// typer.js
//....
const ubahPosisiTidakTerbentur = () => {
  if (this.model.get("x") + word_width > $(window).width()) {
    this.model.set({ x: $(window).width() - word_width });
  }
};
//...
// membuat  text tidak terbentur jika windows di resize
$(window).resize(ubahPosisiTidakTerbentur);
// ....
```

6. Implementasikan sistem score.

Jawab:

Unntuk mengimplementasikan sistem score, saya membuat property dengan nama skor pada model Typer, dengan nilai inisial 0, skor akan direset jika game start dan stop,

```
// ...
 skor: 0,
//  ....
```

nilai skor akan bertambah sebanyak jumlah karakter pada kata yang berhasil diselesaikan

```
if (typed_string.length == string.length) {
  $(this).val("");
  // skor bertambah sebanyak jumlah karakter pada kata jika benar semua kata
  let skor = self.model.get("skor");
  self.model.set({ skor: skor + string.length });
}
```

Agar skor dapat ditampilkan maka saya membuat View dengan variable dengan nama SkorView yang akan melakukan update view nilai ketika skor pada model Typer berubah nilainya

```
// typer.js
// ....
var SkorView = Backbone.View.extend({
  template: _.template(
    `<h3 class='skor'>Skor:
      <% if (skor >= 0) { %>
          <span class="positif"><%= skor %> </span>
      <% } else {%>
          <span class="negatif"><%= skor %> </span>
      <% }%>
    </h3>`
```

```
  ),
  render: function() {
    // agar rerender tidak berlapis
    $(".skor").remove();
    $(this.el).append(this.template(this.model.toJSON()));
    return this;
  },
  initialize: function() {
    this.render();
    this.model.on("change", this.render, this);
  }
});
// ....
```

7. Implementasikan hukuman berupa pengurangan nilai bila salah ketik.

Jawab:

Saya mengimplementasikan hukuman pengurangan skor bila pengguna salah mengetik tiap karakter dengan mengurangi skor dengan angka 1, agar aksi pengurangan skor tidak berulang maka saat pengguna menghapus karakter yang salah perintah pengurangan tidak akan terjadi sampai pengguna memasukan karakter baru pada kolom input.

```
// typer.js
// ....
// skor berkurang -1 jika salah ketik
if (evt.which !== 8) {
  const listKata = words.map(item => item.get("string").toLowerCase());
  const tulisanDiketik = $(this)
    .val()
    .toLowerCase();
  const isSalahKata = !listKata.some(kata => kata.includes(tulisanDiketik));
  let skor = self.model.get("skor");
  if (isSalahKata) {
    skor -= 1;
    self.model.set({ skor });
  }
}
// ....
```